Winning Space Race with Data Science

Hüseyin Furkan Ceran
04/22/2022

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary

- Summary of methodologies
  - Data Collection through API
  - Data Collection with Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Summary of all results
  - Exploratory Data Analysis result
  - Interactive analytics in screenshots
  - Predictive Analytics result

Introduction

- Project background and context

  Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

- Problems you want to find answers

  - What factors determine if the rocket will land successfully?
  - The interaction amongst various features that determine the success rate of a successful landing.
  - What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- The data was collected using various methods
  - Data collection was done using get request to the SpaceX API.
  - Next, we decoded the response content as a Json using .json() function call and turn it into a pandas dataframe using .json_normalize().
  - We then cleaned the data, checked for missing values and fill in missing values where necessary.
  - In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
  - The objective was to extract the launch records as HTML table, parse the table and convert it to a pandas dataframe for future analysis.

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, clean the requested data and did some basic data wrangling and formatting.

- The Jupyter notebook is on the link below :

- GITHUB LINK IPNB

Now let's start requesting rocket launch data from SpaceX API with the following URL:

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

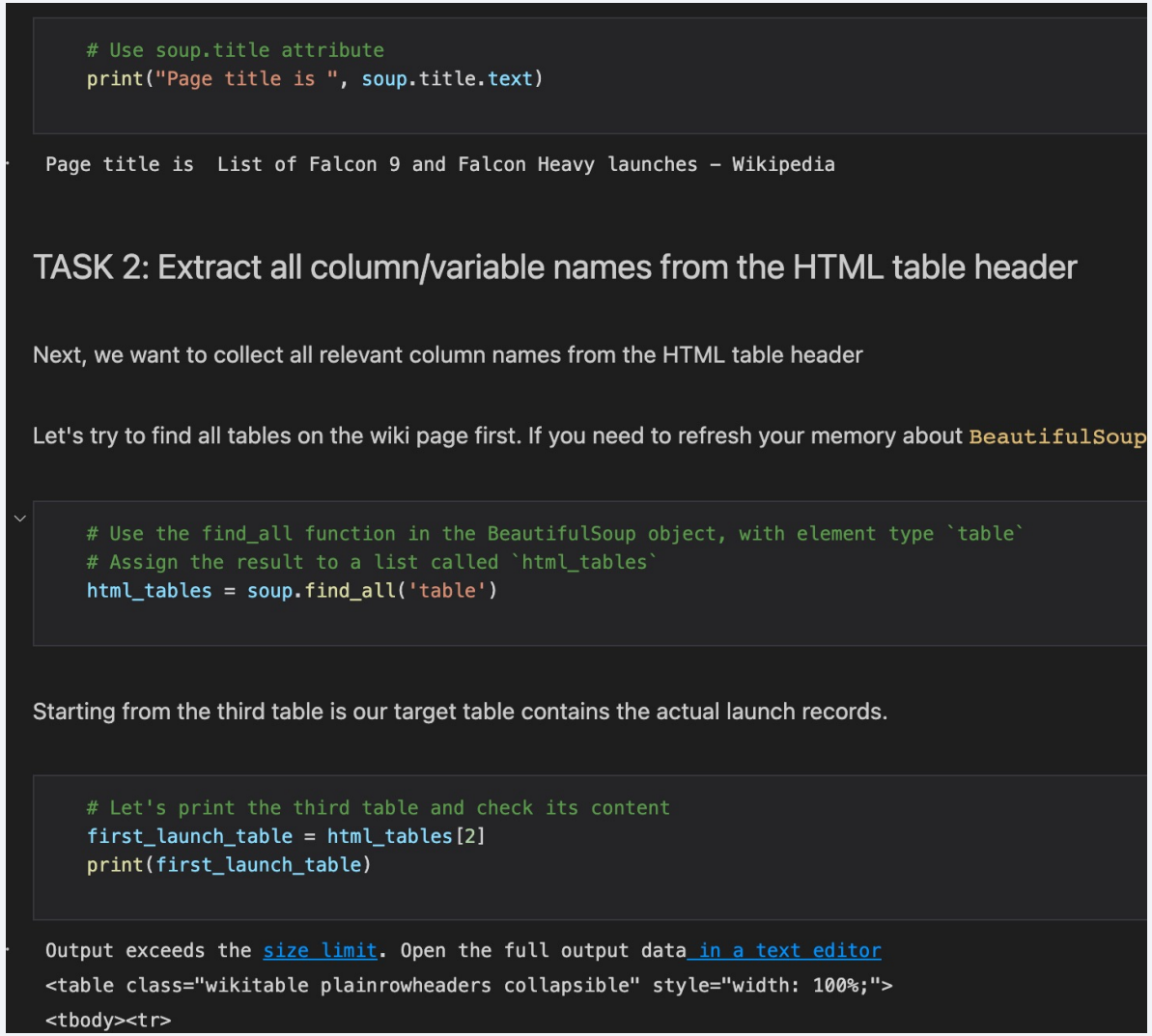
Check the content of the response

```
print(response.content)
```

```
b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ship
{"small":"https://images2.imgbox.com/3c/0e/T8iJcSN3_o.png","large":"https://ima
{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":
v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196
```

Data Collection - Scraping

- We applied web scrapping to webscrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas dataframe.
- The Jupyter notebook is on the link below :
- GITHUB LINK IPNB



```
# Use soup.title attribute
print("Page title is ", soup.title.text)
```

```
Page title is  List of Falcon 9 and Falcon Heavy launches – Wikipedia
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about BeautifulSoup

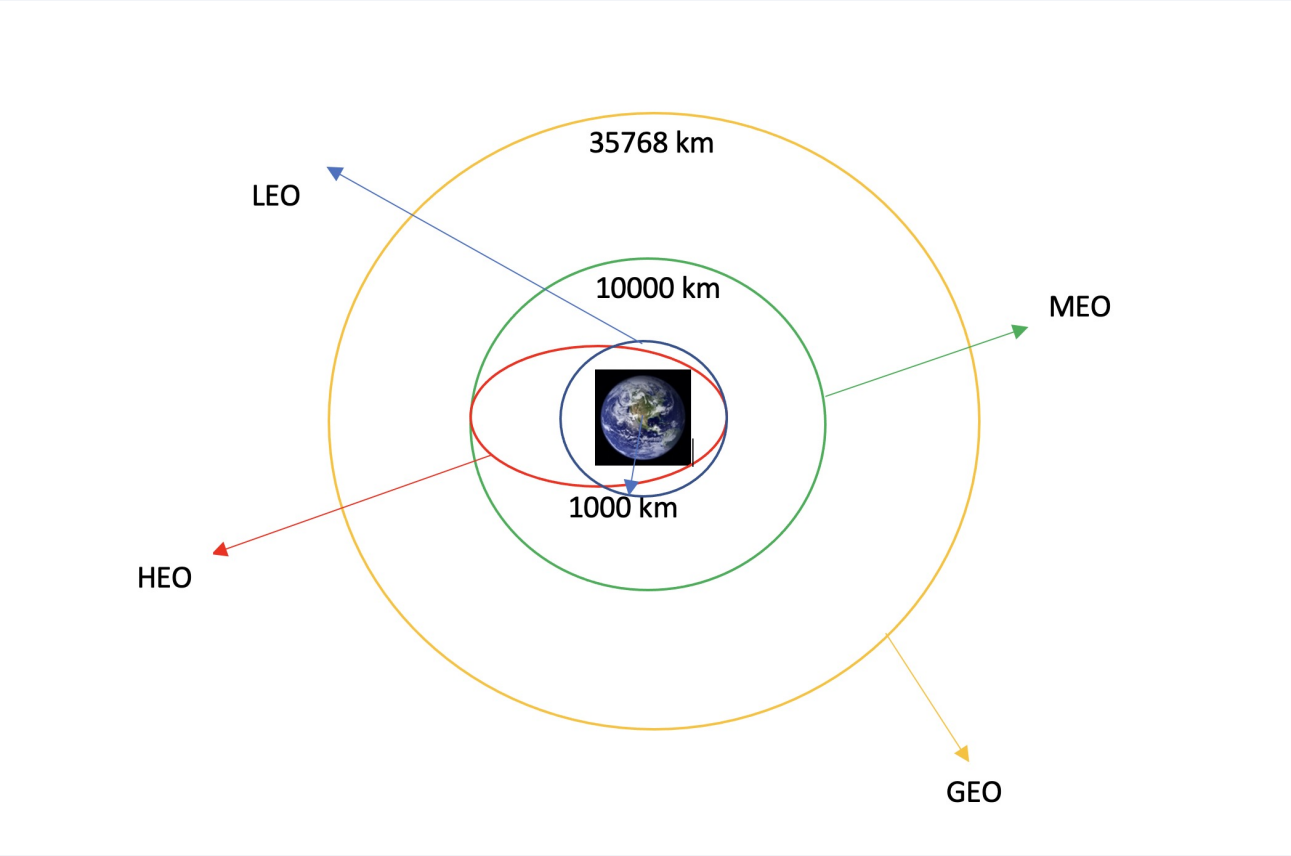
```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```

Starting from the third table is our target table contains the actual launch records.

```
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
Output exceeds the size limit. Open the full output data in a text editor
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
```

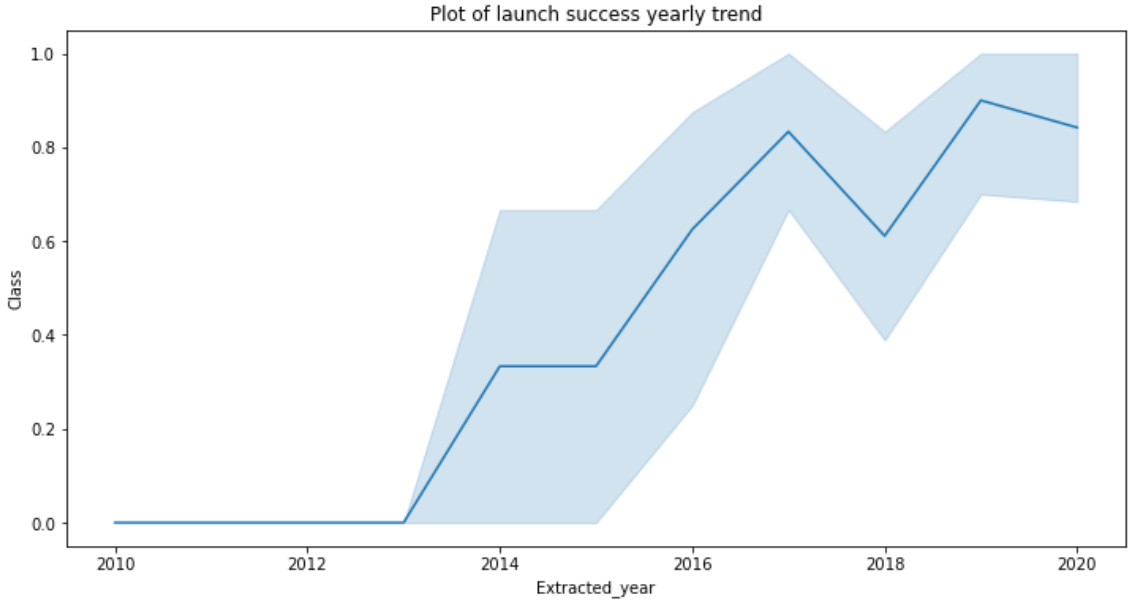
Data Wrangling

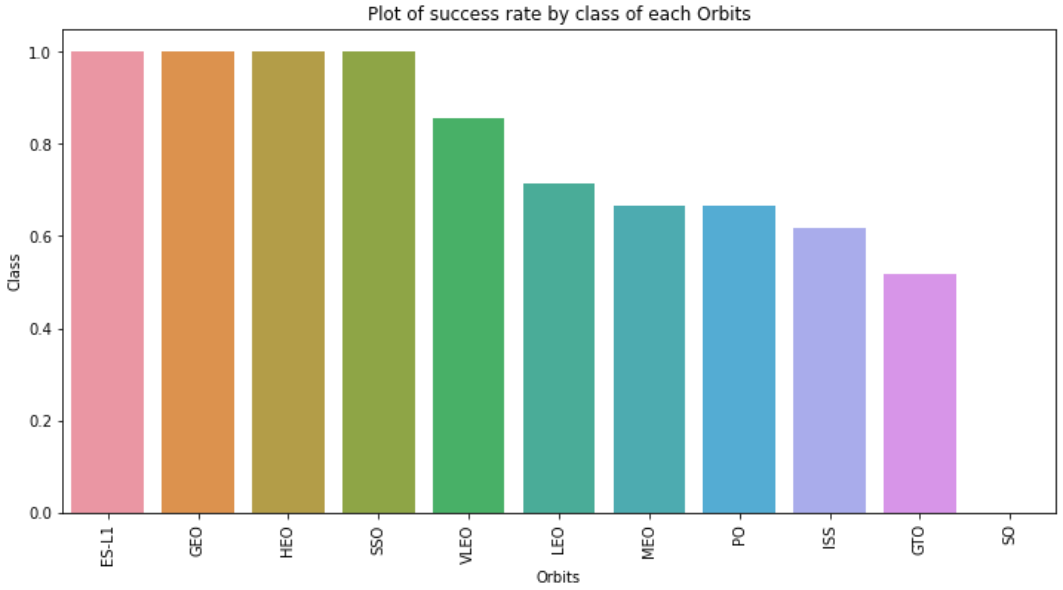


- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbits
- We created landing outcome label from outcome column and exported the results to csv.
- The link to the notebook is :
- GITHUB LINK IPNB

EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, success rate of each orbit type, flight number and orbit type, the launch success yearly trend.





- The link to the notebook is :
- GITHUB LINK IPNB

EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook is :
- GITHUB LINK IPNB

Build an Interactive Map with Folium

- We marked all launch sites, and added map objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to class 0 and 1.i.e., 0 for failure, and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some question for instance:
  - Are launch sites near railways, highways and coastlines.
  - Do launch sites keep certain distance away from cities.

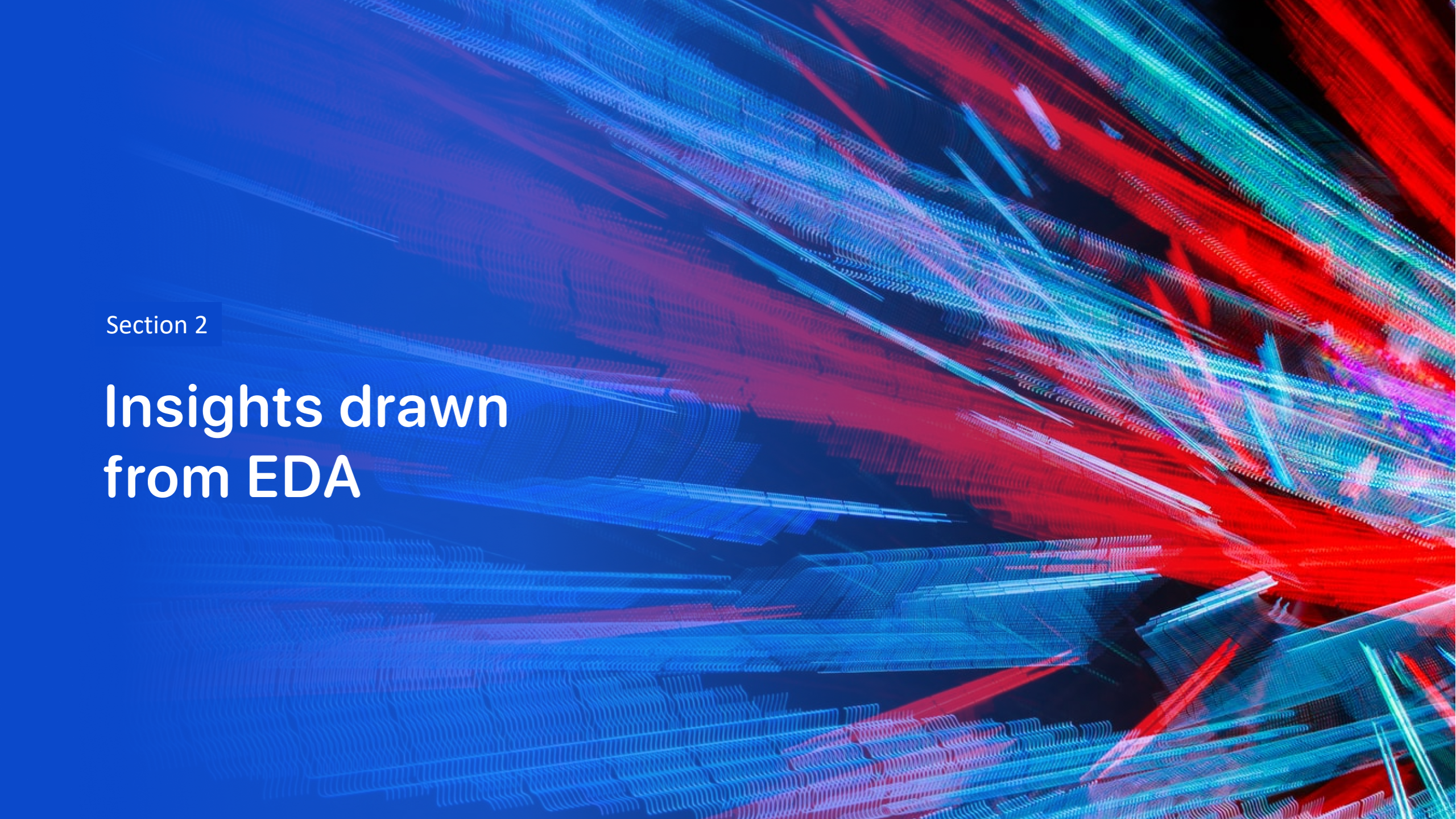

New York
Washington
United States
10
VAFB
SLC-
4E
Phoenix
Los Angeles
46
CCAFS
SLC-
40
The Bahamas
México
La Habana
Cuba
Ciudad
de México
República
Dominicana

Predictive Analysis (Classification)

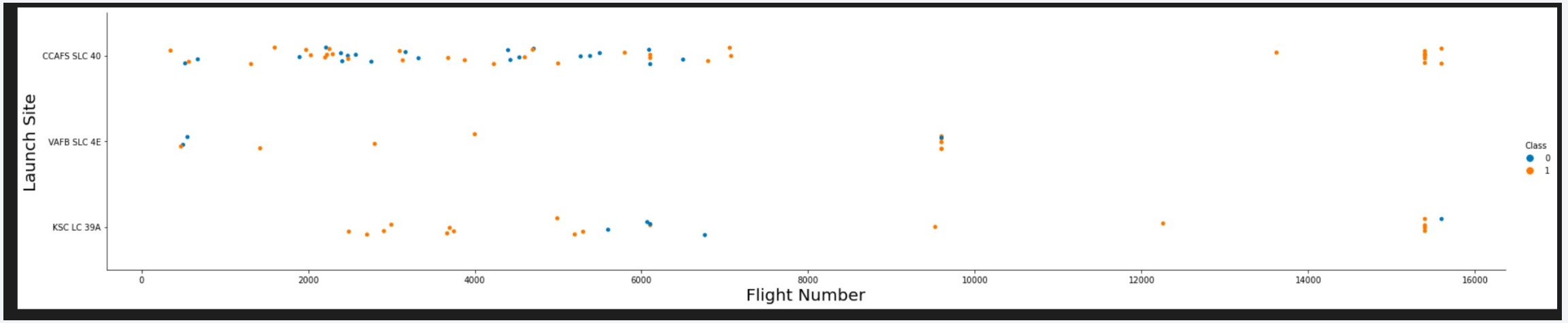
- We loaded the data using numpy and pandas, transformed the data, split our data into training and testing.
- We built different machine learning models and tune different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model, improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is :
- GITHUB LINK IPNB

Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results


Section 2
Insights drawn from EDA

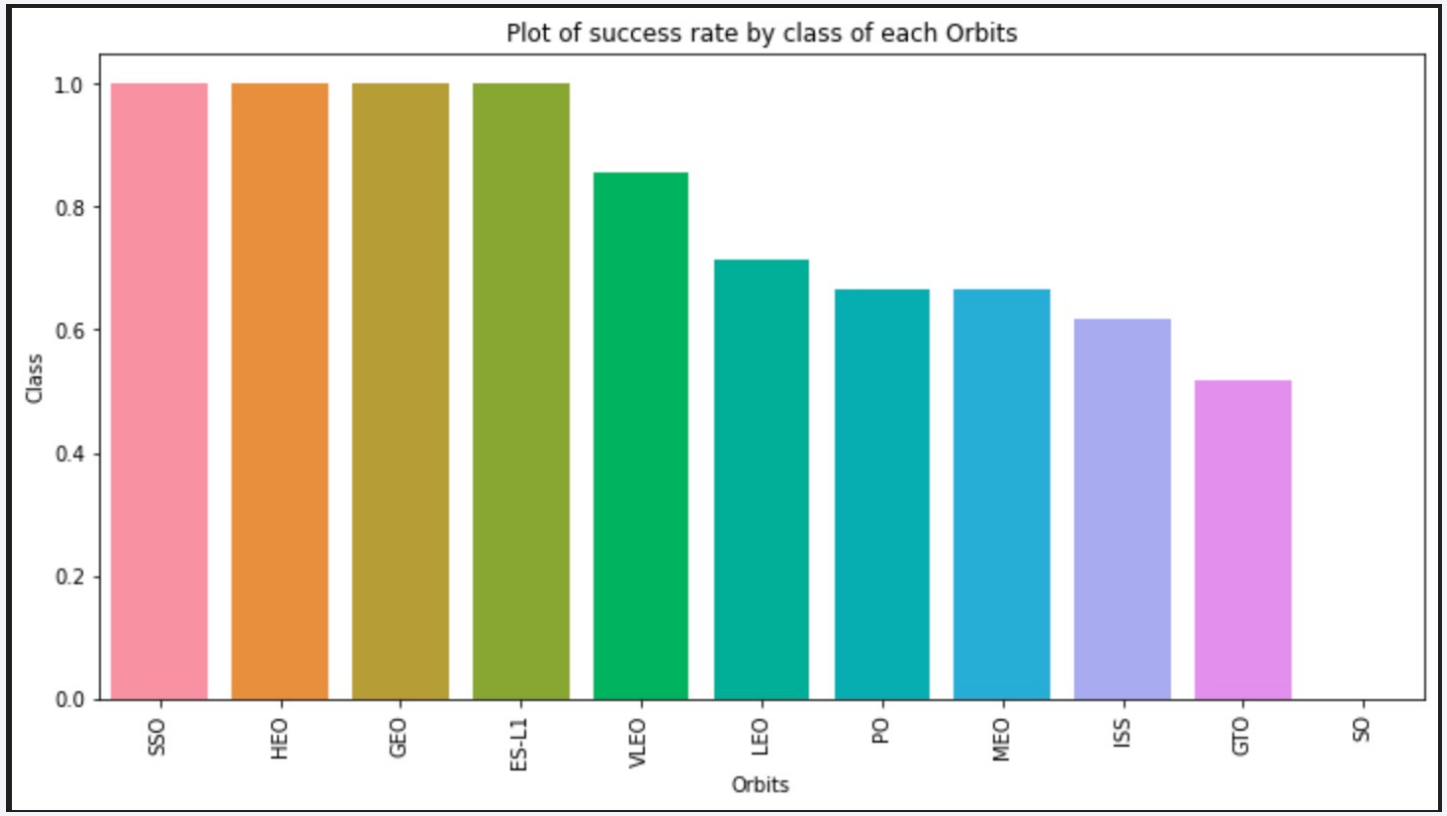
Flight Number vs. Launch Site



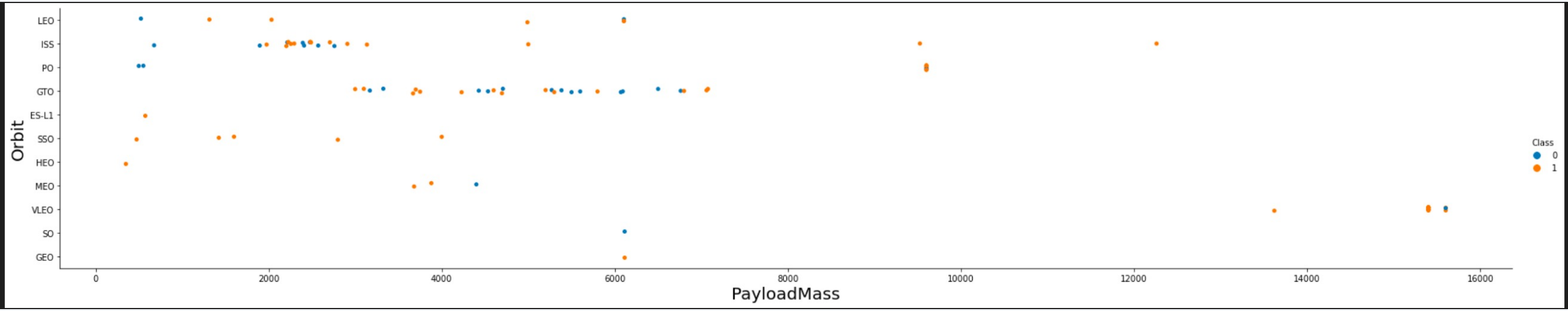
From the plot, we can see that the larger the flight amount at that launch site, the greater the success rate.

Success Rate vs. Orbit Type

- From the plot, we can see that ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



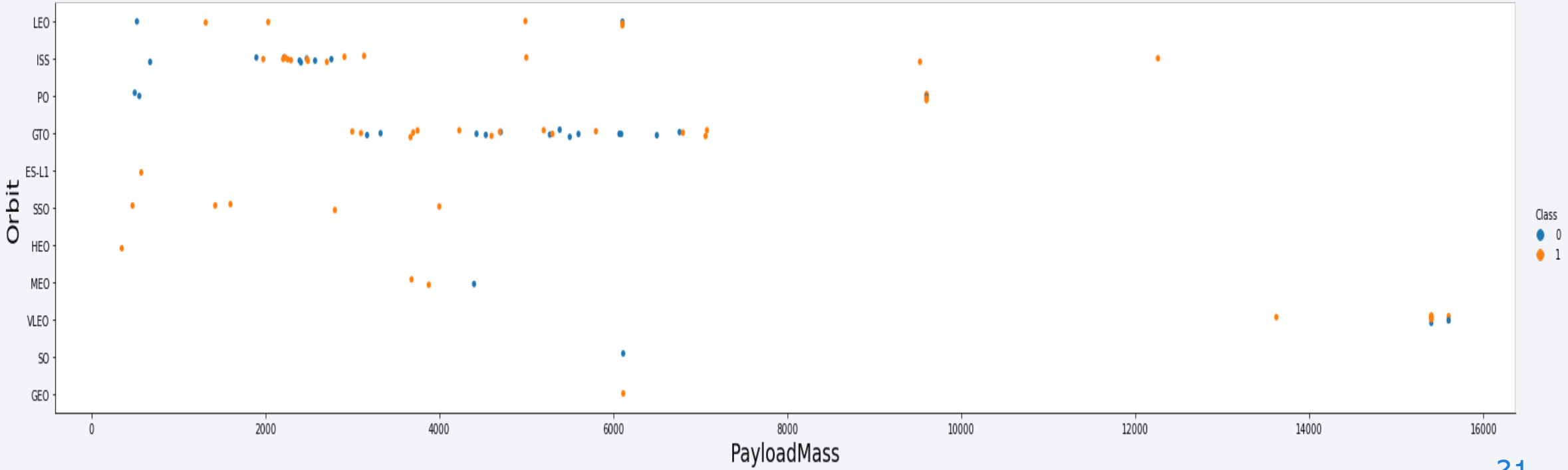
Flight Number vs. Orbit Type



- The plot below shows the Flight Number vs. Orbit type. We observe that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.

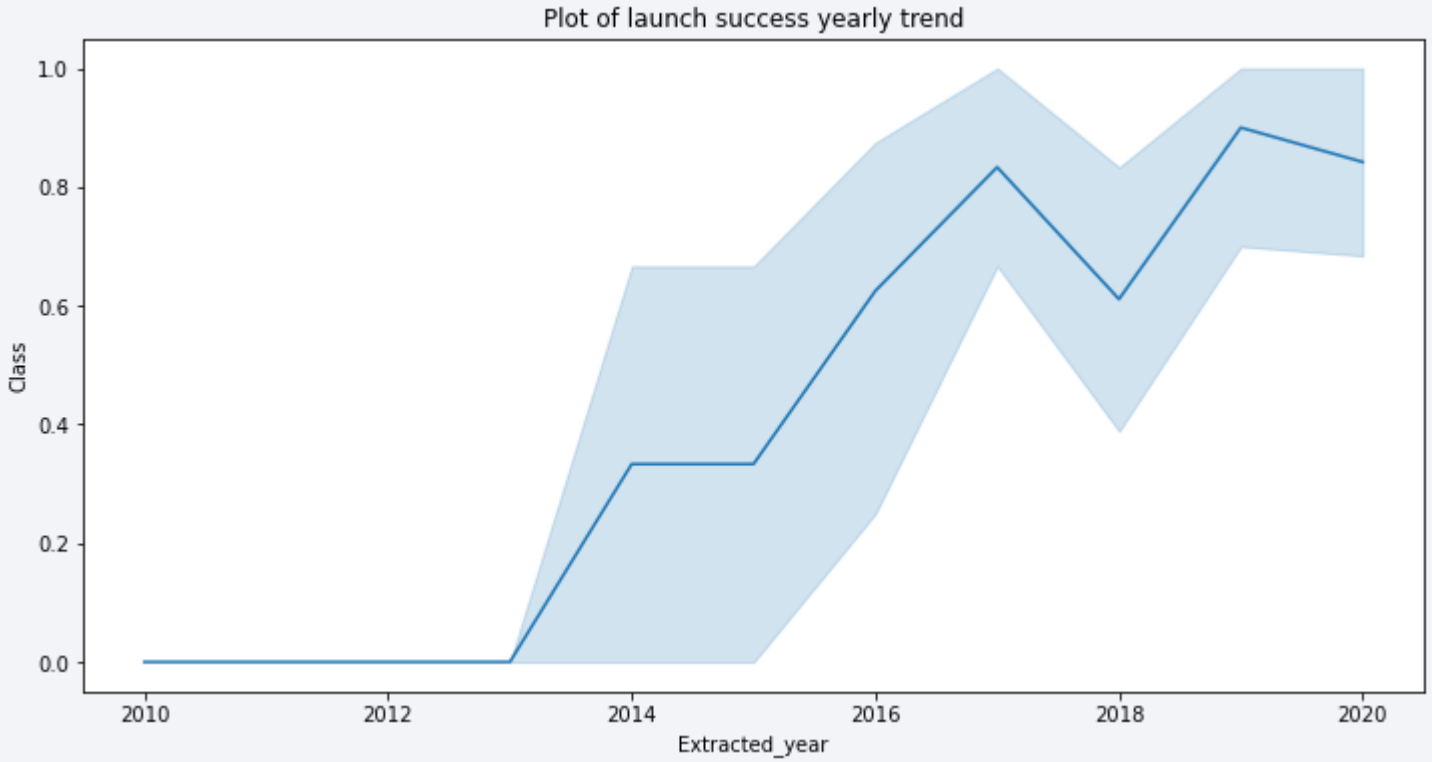
Payload vs. Orbit Type

- We can observe that with heavy payloads, the successful landing are more for PO, LEO and ISS orbits.



Launch Success Yearly Trend

- From the plot, we can observe that success rate since 2013 kept on increasing till 2020.



All Launch Site Names

Task 1

Display the names of the unique launch sites in the space mission

```
%sql SELECT DISTINCT(launch_site)FROM SPACEXTBL
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

- We used the query above to display the names of the unique launch sites.

Launch Site Names Begin with 'CCA'

Task 2

Display 5 records where launch sites begin with the string 'CCA'

```
%sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE '%CCA%' LIMIT 5
```

Python

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-12	22:41:00	F9 v1.1	CCAFS LC-40	SES-8	3170	GTO	SES	Success	No attempt

- We used the query above to display 5 records where launch sites begin with CCA

Total Payload Mass

- We calculated the total payload carried by boosters from NASA

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

```
%sql SELECT SUM(payload_mass__kg_) as Total_PayloadMass FROM SPACEXTBL WHERE customer =  'NASA (CRS)'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

total_payloadmass
22007

Average Payload Mass by F9 v1.1

- We calculated the average payload mass carried by booster version F9 v1.1

Task 4

Display average payload mass carried by booster version F9 v1.1

```
%sql SELECT AVG(payload_mass__kg_) AS Avg_PayloadMass FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

avg_payloadmass
3676

First Successful Ground Landing Date

- We observed that the dates of the first successful landing outcome on ground pad was 2017-01-05

Task 5

List the date when the first successful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql SELECT MIN(DATE) AS first_success FROM SPACEXTBL WHERE landing__outcome LIKE 'Success (ground pad)'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

first_success
2017-01-05

Successful Drone Ship Landing with Payload between 4000 and 6000

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
%sql SELECT booster_version FROM SPACEXTBL WHERE landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4000 AND payload_mass__kg_ < 6000
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

booster_version
F9 FT B1022
F9 FT B1031.2

- We used the **WHERE** clause to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

- We used wildcard like '%' to filter for **WHERE** MissionOutcome was a success or a failure.

Task 7

List the total number of successful and failure mission outcomes

```
%sql SELECT COUNT(mission_outcome) AS Number_Success FROM SPACEXTBL WHERE mission_outcome LIKE '%Success%'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

number_success
45

Boosters Carried Maximum Payload

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%sql SELECT booster_version, payload_mass__kg_ FROM SPACEXTBL WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL) ORDER BY booster_version
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1058.3	15600
F9 B5 B1060.2	15600

- We determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function.

2015 Launch Records

- We used a combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015

Task 9

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%sql SELECT booster_version, launch_site, landing__outcome FROM SPACEXTBL WHERE landing__outcome LIKE 'Failure (drone ship)' AND Date BETWEEN '2015-01-01' AND '2015-12-31'
```

```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

booster_version	launch_site	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

Task 10

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
%sql SELECT landing__outcome, COUNT(landing__outcome) FROM SPACEXTBL WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing__outcome ORDER BY COUNT(landing__o
```

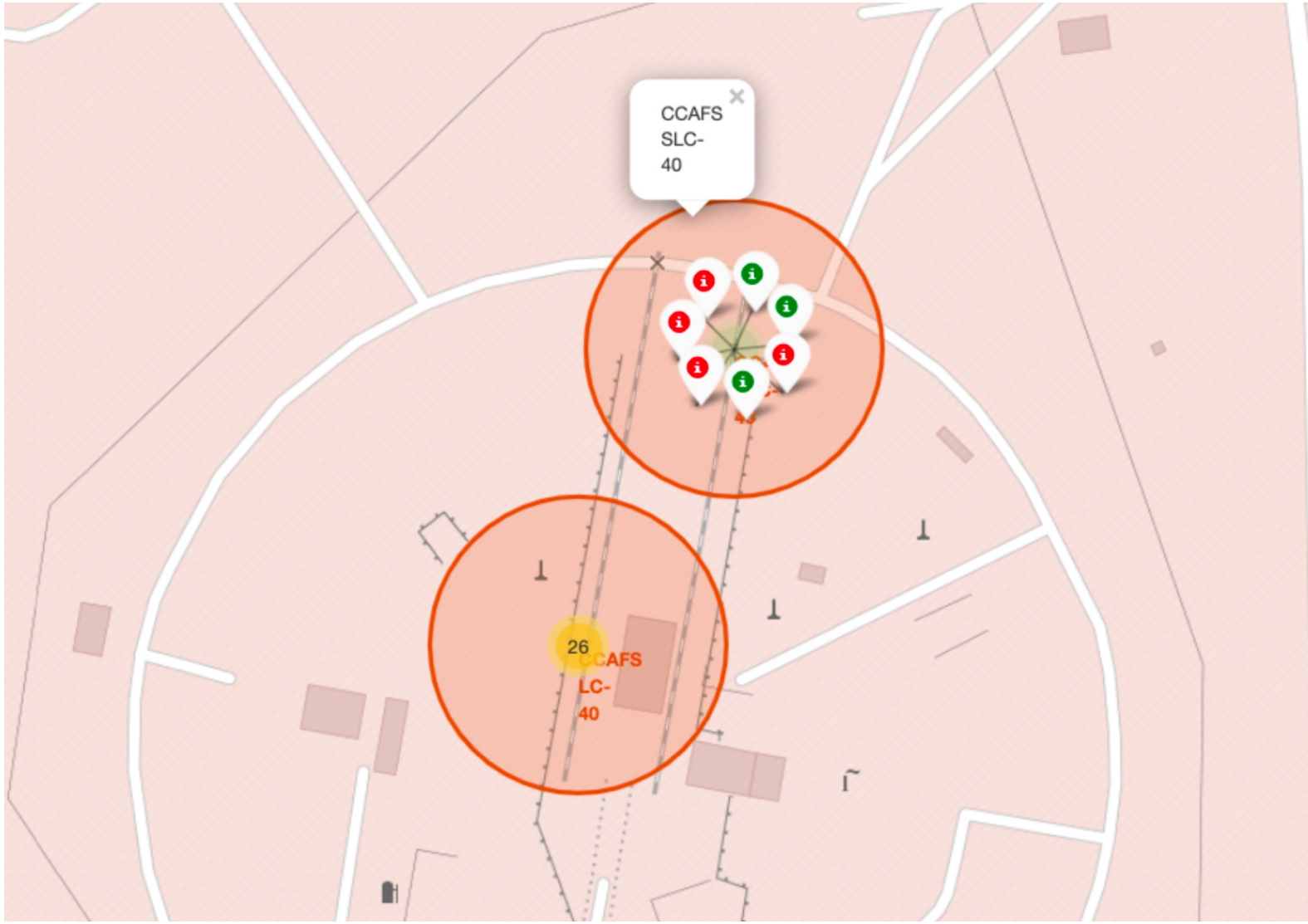
```
 * ibm_db_sa://jxz81713:***@8e359033-a1c9-4643-82ef-8ac06f5107eb.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30120/bludb
Done.
```

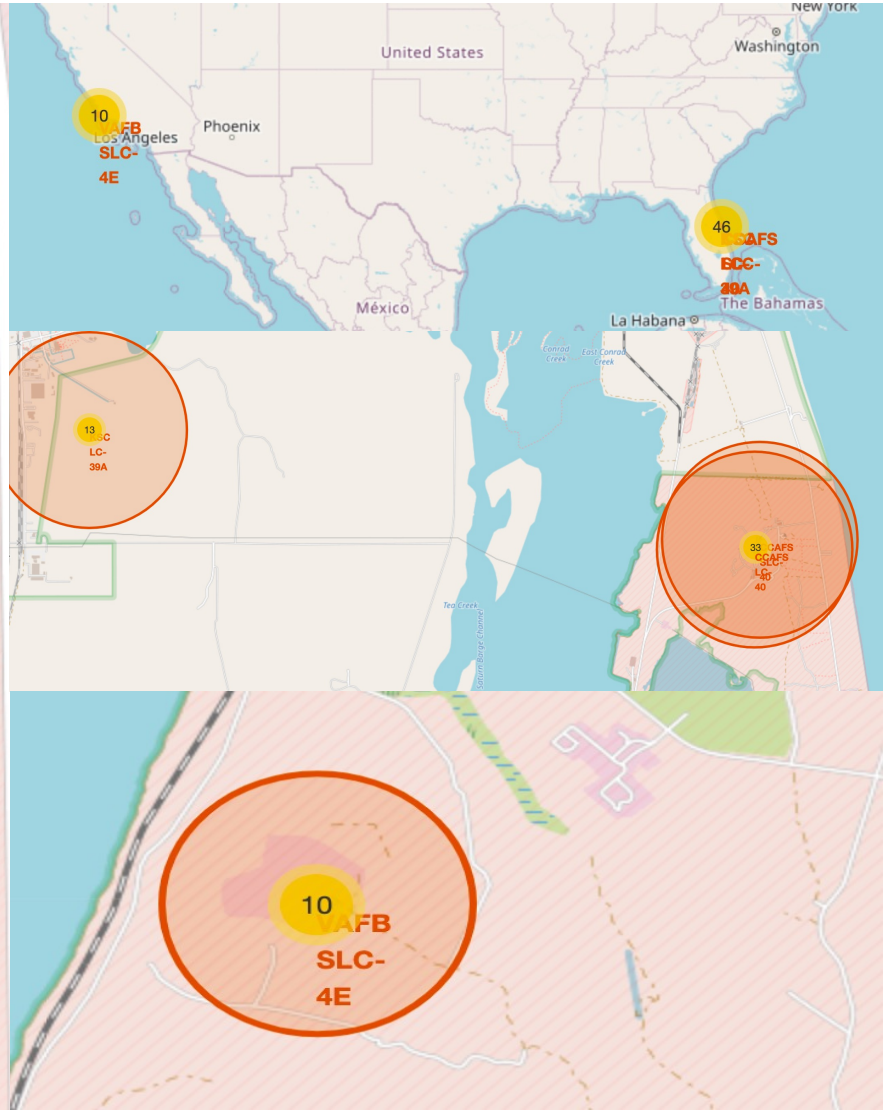
landing__outcome	2
No attempt	7
Failure (drone ship)	2
Success (drone ship)	2
Success (ground pad)	2
Controlled (ocean)	1
Failure (parachute)	1

- We used WHERE and GROUP BY and ORDER BY.

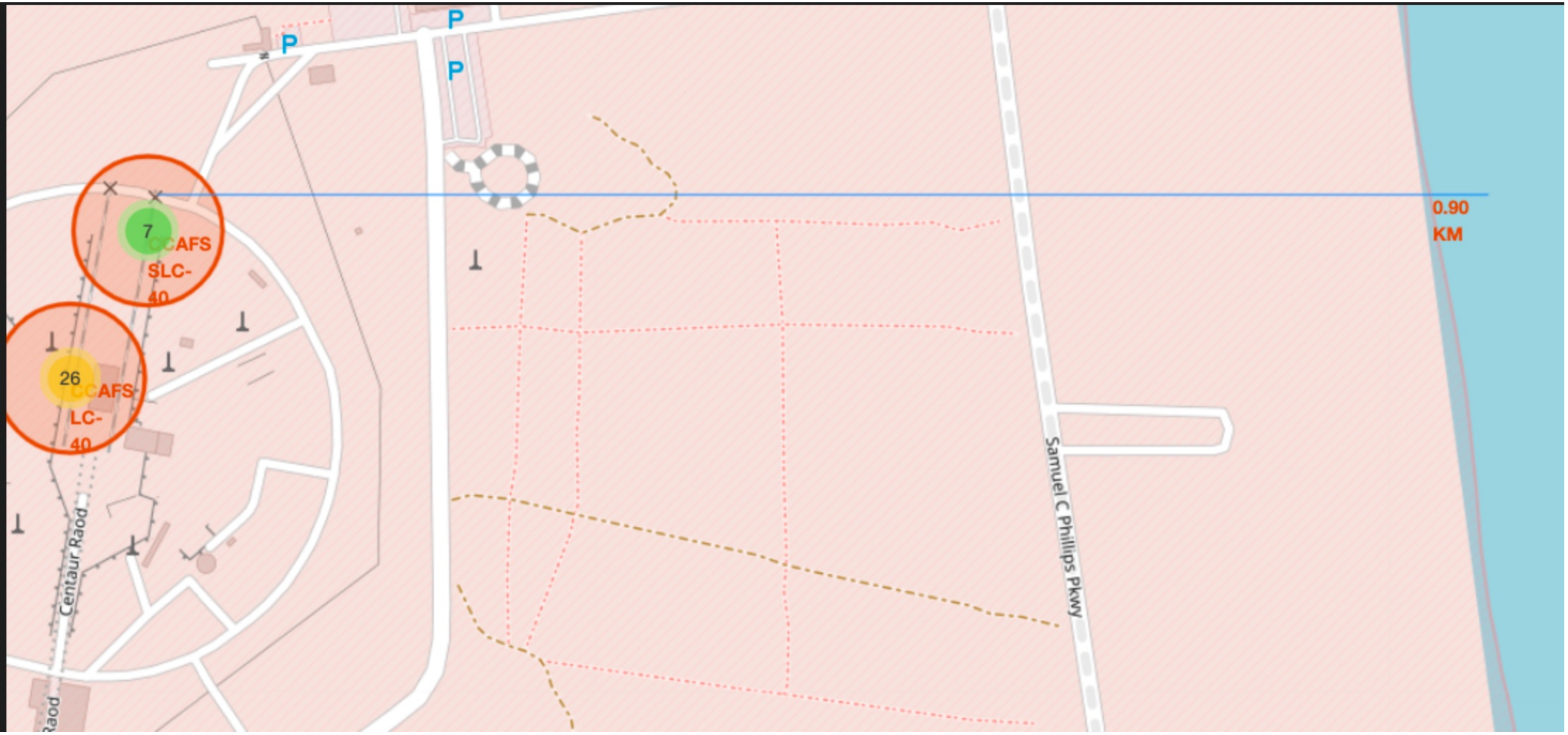

Section 3
Launch Sites
Proximities Analysis

Markers showing launch sites with color labels

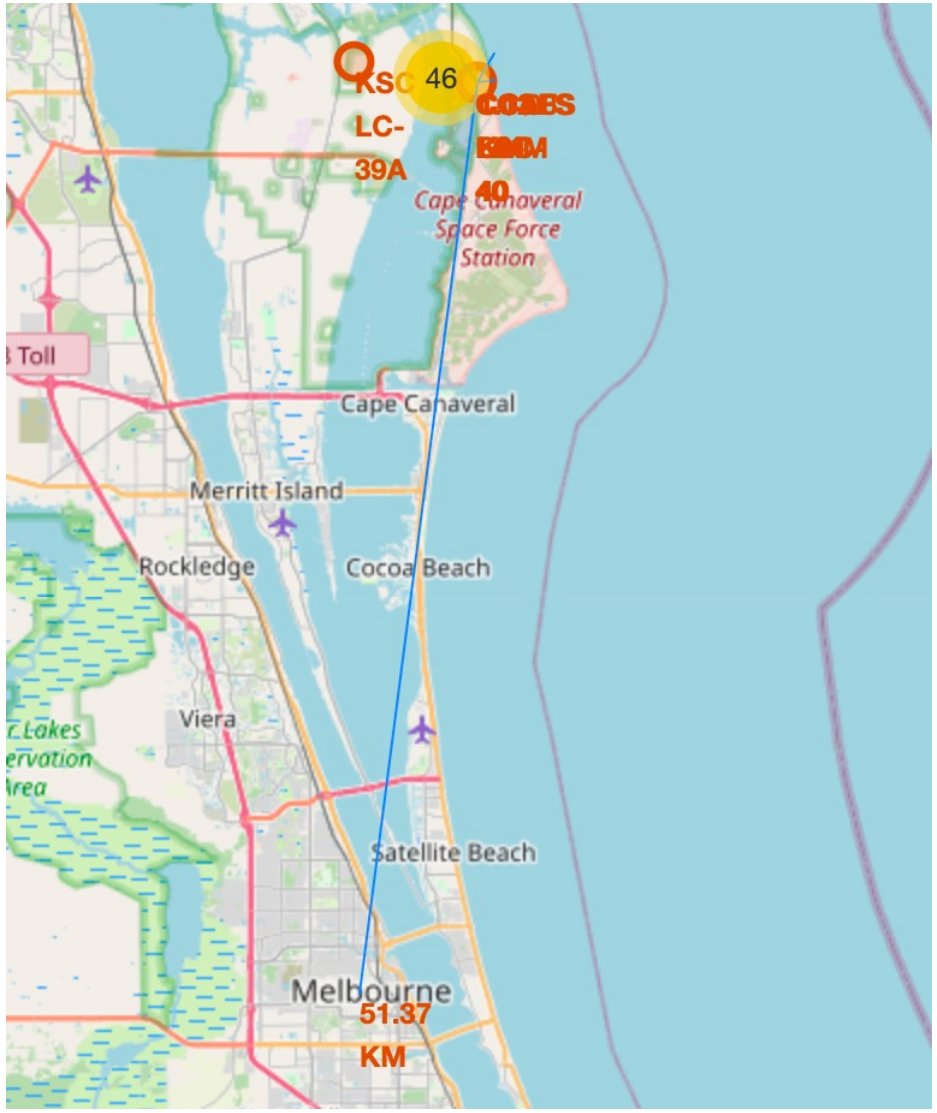


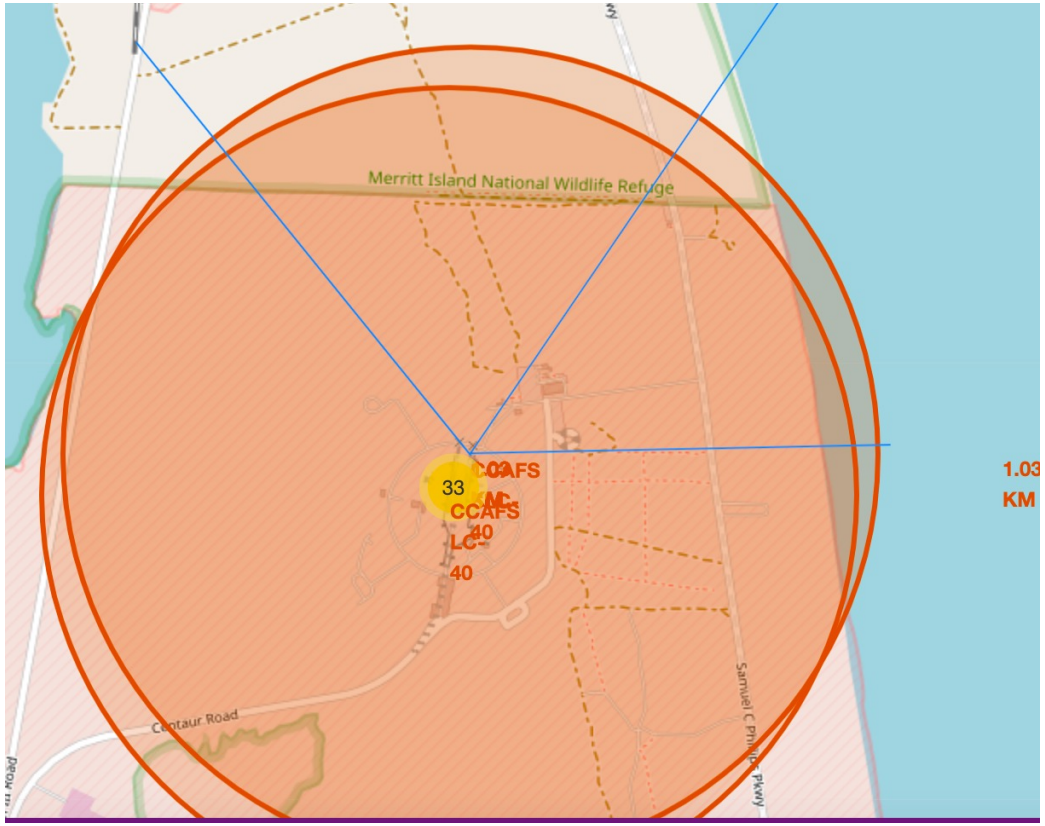


Launch Site distance to landmarks



Launch Site distance to landmarks





Section 5

Predictive Analysis (Classification)

Classification Accuracy

```
ml = {'LogisticRegression':logreg_cv.best_score_, 'SVM':svm_cv.best_score_,'Tree':tree_cv.best_score_,'KNN':knn_cv.best_score_}

best = max(ml, key=ml.get)
print('Best ML Training Algorithm is',best,'with a score of',ml[best])
if best == 'LogisticRegression':
    print('Best method is :',logreg_cv.best_
if best == 'SVM':
    print('Best method is :',svm_cv.best_params_)
if best == 'Tree':
    print('Best method is :',tree_cv.best_params_)
if best == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
```

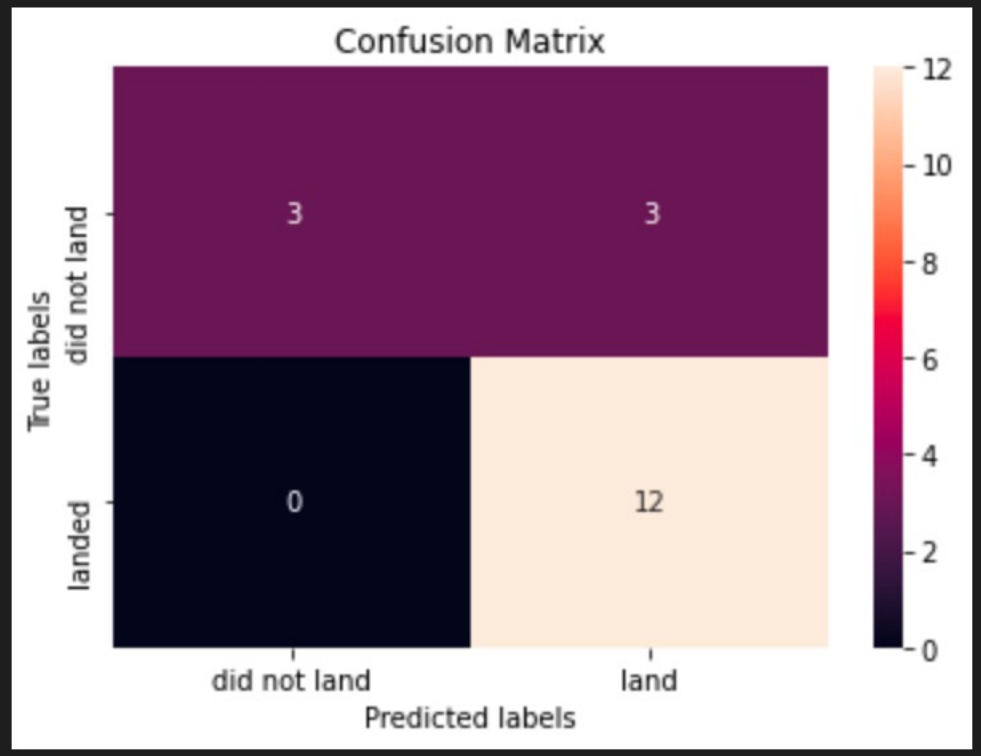
Yükleniyor...

```
Best ML Training Algorithm is Tree with a score of 0.8892857142857145
Best method is : {'criterion': 'gini', 'max_depth': 6, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'splitter': 'random'}
```

- The best model is Decision Tree with an Accuracy of 88.92%

Confusion Matrix

```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(y_test,yhat)
```



- The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.

Conclusions

- This project and analysis allow us to conclude the following
:
- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate is in continuous increase since 2013.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the most success rate.
- KSC LC-39A had the most successful launches of any sites.
- The Decision tree classifier is the best machine learning algorithm for this task.

Thank you!